TRAINING MODULE

# Poka-Yoke Hardening

*Mistake-proofing applied to the data itself, not just the line*

A working module for engineers with 1–2 years on the floor. By the end you will be able to explain true mistake-proofing, and recognize why a study's own data template deserves the same discipline as a fixture on the shop floor.

*Grounded in Shigeo Shingo's Poka-Yoke and Zero Quality Control (ZQC) principles*

BEFORE WE START

# What You Will Be Able to Do After This Module

1. Define Poka-Yoke correctly, and explain how it differs from ordinary error detection.
2. Explain why THIS tool applies Poka-Yoke to a data template, not a physical process — and why that still counts.
3. Describe all four hardening mechanisms this tool applies, and what error each one prevents.
4. Explain the CT formula bug this tool fixes, and calculate the difference by hand.
5. Recognize the two most common mistakes junior IEs make when they first think about mistake-proofing.

# A Mistake You Cannot Physically Make

Most quality thinking asks “how do we CATCH the error?” Poka-Yoke asks a stricter question:

## DETECTION

“Inspect the part after assembly to catch a missing component.” The mistake already happened — you're just finding it.

## POKA-YOKE (MISTAKE-PROOFING)

“Design the fixture so a mis-oriented part physically cannot be inserted at all.” The mistake becomes impossible, not just catchable.

**Shingo's own term for the ideal: Zero Quality Control (ZQC) — not zero inspection effort, zero DEFECTS, because the mistake was designed out entirely.**

THE CORE IDEA

# The Same Principle, Aimed at Your Own Workbook

Poka-Yoke usually protects a physical assembly. This tool asks: what if the thing that needs protecting is the Production Study itself?

⚠ An operator accidentally deletes a row mid-study — every downstream formula silently shifts.

⚠ Someone overtypes a formula cell with a number, and it never recalculates again.

⚠ A new IE isn't sure what a column expects — types the wrong format, breaks a chain of formulas.

⚠ A per-piece station's cycle time gets compared to Takt as if it were a raw batch number.

**Every one of these is a real, common way a study's data quietly breaks — and every one of the four mechanisms on the next slides makes exactly one of them physically harder to do by accident.**

WHERE THIS METHOD COMES FROM

# Inspection Should Prevent Defects, Not Just Find Them

Shingo's insight, developed at Toyota supplier plants, was that source inspection combined with a simple, cheap physical or procedural device could make a defect literally impossible to create — not just likely to be caught. A pin that won't fit a part backward. A weight sensor that won't let a press cycle without both screws seated. The device does the checking, permanently, without relying on human attention.

*Sheet protection and locked formula cells are the exact same idea, aimed at a spreadsheet instead of a fixture.*

**SHIGEO SHINGO**

*Zero Quality Control (ZQC)*

THE METHOD, IN ONE PICTURE

# Four Mechanisms, One Workbook

1

### TWO-COLOR CODING

Green = editable. Grey = locked. No guessing.

2

### STRUCTURAL PROTECTION

Row/column insert & delete blocked. Cell edits still allowed.

3

### INSTRUCTIONAL COMMENTS

Every editable header explains exactly what to type.

4

### FORMULA FIXES

CT logic corrected — MAX not SUM, per-piece not per-pass.

*The rest of this module walks each one with a real worked example.*

1

MECHANISM 1

# Two-Color Coding

A new IE should never have to guess which cell is safe to type into — the color already tells them.

*Applied to every cell in the workbook, not just a sample — consistency is the whole point.*

THE RULE

**GREEN (editable)**

Raw data cells — element description, C1/C2/C3 readings, station name.

**GREY (locked)**

Anything starting with “=” — formula-driven results, computed totals.

HOW IT DECIDES

**is_formula(cell) checks if the value starts with “=” — that single test decides green vs. grey for every cell, every time this tool runs. No manual classification, no cells missed.**

2

MECHANISM 2

# Structural Protection

The key design choice: block STRUCTURE changes, not DATA changes. A full sheet lock would make the workbook useless.

*Right-click Insert/Delete on a row or column literally greys out — not a warning dialog, a physical impossibility.*

*Mechanism 2 — Structural Protection*

WHY NOT JUST LOCK EVERYTHING?

### FULL LOCK (too far)

Nobody could enter a single time reading. The workbook becomes decoration, not a tool.

### NO PROTECTION (too little)

One accidental “Delete Row” mid-study silently shifts every formula reference below it.

### STRUCTURAL PROTECTION (this tool)

Cell VALUES stay editable. Row/column STRUCTURE stays fixed. Both real risks addressed, neither over-corrected.

3

MECHANISM 3

# Instructional Comments

Placed on the header cell of every editable column — visible the moment someone hovers, before they type anything wrong.

*Covers linkage rules too, like Start Time's shared-clock behavior — not just "what goes here."*

EXAMPLE — COMMENT ON THE "START TIME" HEADER

*"Enter the clock time this element begins. This column shares a clock with every other station's Start Time — changing one shifts the reference for all of them. Format: HH:MM:SS."*

WHAT THIS PREVENTS

*Without this note, a new IE could reasonably assume Start Time is local to just their station — and be genuinely confused when editing one field shifts numbers on a completely different sheet.*

**Comment author is always "IE System" — consistent, traceable, never mistaken for a person's personal note left in the file.**

MECHANISM 4 — A REAL BUG, ACTUALLY FIXED

# L4 = MAX(End Time), Never SUM(End Time)

Five elements run in sequence. Their End Time column is CUMULATIVE — each one already includes everything before it:

| Element | Duration | End Time (cumulative) |
|---|---|---|
| E1 | 8s | 8s |
| E2 | 7s | 15s |
| E3 | 9s | 24s |
| E4 | 9s | 33s |
| **E5** | **7s** | **40s** |

**WRONG: SUM(End Time)**

**8+15+24+33+40 = 120s**
**— 3× inflated, meaningless**

**RIGHT: MAX(End Time)**

**MAX(8,15,24,33,40) = 40s**
**— the actual station cycle time**

**This is the exact same station-CT logic from the Bottleneck module — the last element's END TIME already IS the station's cycle time. And it cascades: Q3 “Prod/hr” and Index column G are keyed off the corrected per-piece M4, not the raw per-pass L4, with a visible note wherever a station's pcs/cycle > 1 — the exact per-piece correction from the SWCS module.**

THE POKA-YOKE APPLIED TO THE POKA-YOKE TOOL ITSELF

# Idempotent by Design — Never Accumulates

Every step recomputes fresh from the sheet's own current data, every single run:

**Run 1**
**(new stations added)**

Same styling. Same protection. Same fixed formulas. No duplicate comments, no layered fills.

=

**Run 2**
**(re-applied)**

Same styling. Same protection. Same fixed formulas. No duplicate comments, no layered fills.

=

**Run 3**
**(re-applied again)**

Same styling. Same protection. Same fixed formulas. No duplicate comments, no layered fills.

**Why this matters: a hardening tool that accumulates side effects every time it's run is itself a mistake waiting to happen — duplicate comments piling up, fills applied on top of fills. This tool holds itself to its own standard.**

**A timestamped BACKUP file is still written before every run — idempotent doesn't mean reckless. Belt AND suspenders.**

BEFORE YOU GO TO THE FLOOR

# Five Mistakes That Give Junior IEs Away

✗ Thinking Poka-Yoke only applies to physical fixtures — the same discipline protects any place a costly mistake can be made.

✗ Assuming sheet protection means the workbook can't be edited at all — it's structure-blocked, not data-blocked.

✗ Skipping the BACKUP file check before re-running this tool — idempotent design still isn't a reason to skip due diligence.

✗ Confusing the green/grey coding for a style choice — it's a functional signal, not decoration.

✗ Manually re-typing over a locked formula cell using unprotect tricks — defeating the exact mistake-proofing this tool exists to provide.

FROM PENCIL-AND-PAPER TO THE SOFTWARE

# What apply_poka_yoke.py Does For You

All four mechanisms, applied across the entire workbook in one pass — safely, every time.

| | | |
|---|---|---|
| 1 | **Colors every cell correctly** | is_formula() decides green vs. grey automatically, across every sheet, every column. |
| 2 | **Applies structural protection** | Row/column insert & delete blocked; cell edits stay open — no manual per-sheet setup. |
| 3 | **Writes instructional comments** | Every editable header gets a comment explaining what to type and any linkage rules, authored consistently as “IE System.” |
| 4 | **Fixes the CT formulas** | L4 = MAX not SUM, Q3 and Index column G keyed off per-piece M4, visible notes wherever pcs/cycle > 1. |
| 5 | **Writes a timestamped backup first** | Production_Study_Master_BACKUP_<timestamp>.xlsx, every single run, before anything else happens. |

KEY TAKEAWAYS

# Recap — Say This Back In Your Own Words

- Poka-Yoke means making a mistake physically impossible — not just easier to catch after it happens.
- The same discipline protects a spreadsheet as well as a fixture — this tool proves it, aimed at your own data.
- Structural protection blocks row/column changes while leaving cell values fully editable — both real risks handled.
- A real bug got fixed here: station CT is MAX(End Time), never SUM — the same logic you already know from Bottleneck and SWCS.

## TALK IT THROUGH — 3 QUESTIONS FOR YOUR NEXT LINE WALK

1. Where else in your own workflow could a Poka-Yoke idea apply — outside the shop floor entirely?
2. Have you ever seen a spreadsheet formula silently break from an accidental row delete? What would structural protection have prevented?
3. Pick a formula in your own Production Study — can you explain, cell by cell, whether it should be MAX, SUM, or something else, and why?